Dependencies between modularity metrics towards improved modules

Zubeida Casmod Khan[1,2] and C. Maria Keet[1]

[1] Department of Computer Science, University of Cape Town, South Africa
mkeet@cs.uct.ac.za
[2] Council for Scientific and Industrial Research, Pretoria, South Africa
zkhan@csir.co.za

Abstract. Recent years have seen many advances in ontology modularisation. This has made it difficult to determine whether a module is actually a good module; it is unclear which metrics should be considered. The few existing works on evaluation metrics focus on only some metrics that suit the modularisation technique, and there is not always a quantitative approach to calculate them. Overall, the metrics are not comprehensive enough to apply to a variety of modules and it is unclear which metrics fare well with particular types of ontology modules. To address this, we create a comprehensive list of module evaluation metrics with quantitative measures. These measures were implemented in the new Tool for Ontology Module Metrics (TOMM) which was then used in a testbed to test these metrics with existing modules. The results obtained, in turn, uncovered which metrics fare well with which module types, i.e., which metrics need to be measured to determine whether a module of some type is a 'good' module.

1 Introduction

A number of techniques for ontology modularisation have been proposed in recent years, such as traversal methods [18], locality-based extraction [9], and partitioning [2, 4]. There also have been attempts at analysing which types of modules exist [1], and which types are useful for which purpose, such as that high-level abstraction modules are used for comprehension [15]. There is, however, a disconnect between the two. For instance, if a modeller wants to reuse, say, only the branch of 'social objects' from the DOLCE foundational ontology for an ontology about e-government, then how does the modeller know that the module extracted from DOLCE is a good module, and which one of the modularisation techniques creates the module of the 'best' quality? In fact, it is even unclear how the quality of an ontology module could or should be measured. There are a few studies on evaluation of ontology modules, which focus on a few metrics, such as size, cohesion, coupling, correctness, and completeness [22, 24, 27], however some of them do not have a defined formula to measure them e.g., intra-module distance [4]. The metrics are not comprehensive enough to apply to the 14 types of modules [15] that exist [11].

Metrics such as size do not fare well with modules created using locality-based techniques [12], while completeness and correctness do not measure well with partition-based modules [22]. This suggests that only specific metrics would be applicable for some type of module to assess the quality of an ontology module. To the best of our knowledge, no one has filled this knowledge gap of how evaluation metrics relate to modules to reveal the module quality.

To solve these problems, we take both existing metrics, where for those that do not have a computable component, we devise a new function, and add three more metrics, totalling to 16. To examine their usefulness, we implemented these metrics into one evaluation tool, the Tool for Ontology Modularity Metrics (TOMM) and examined 189 ontology modules on these metrics. TOMM can be downloaded from `http://www.thezfiles.co.za/Modularity/TOMM.zip`. These results from TOMM generated insight into the expected values for evaluation metrics for the different types of modules. We have evidence-based insight about which metrics fare well with which module types. For instance, ontology matching modules fare well with a mix of structural, relational, and information hiding metrics. This insight helps the ontology developer to determine whether a module is of good quality or not.

The remainder of the paper is structured as followed. Related works are summarised in Section 2, followed by the evaluation metrics for modules in Section 3. The software, TOMM, experimental evaluation, and use-cases is presented in Section 4, and a discussion in Section 5. Lastly, we conclude in Section 6.

2 Related works

In order to solve the problem of insufficient modularity metrics, we look at the existing work. To start with we use a definition of a module by Khan and Keet stating that a module M is a subset of a source ontology or module M is an ontology existing in a set of modules such that, when combined, make up a larger ontology [15]. In addition, Khan and Keet created a framework for ontology modularity, aimed at guiding the modularisation process [15]. It consists of dimensions for modularity such as use-cases, techniques, types, and properties. The dimensions have been linked to reveal dependencies, and to annotate modules with additional information. It does not include module evaluation. Other works do consider this. For instance, Pathak et. al [22] identified main properties that modules need to satisfy, such as size, correctness, completeness, and evaluated them using existing tools, noting that module correctness is satisfied by most techniques and that completeness and size are difficult to satisfy. They also observed that the logic-based approaches tend to result in modules where completeness is achieved while the graph-based approaches generate modules of smaller size that are not logically complete.

Schlicht and Stuckenschmidt [24] created a set of structural criteria for ontology modules, including connectedness, size, and redundancy of representation, and use quantitative functions to formally measure each criteria value. They argue that structural criteria have an effect on efficiency, robustness and main-

tainability for the application of semantics-based peer-to-peer systems. They evaluated the SWOOP and PATO modularity tools on their structural criteria, observing that SWOOP favours modules with a good connectedness over modules with suitable size values, whereas with PATO, a threshold value could be selected such that when the threshold value is increased, so did the size suitability of the module to the detriment of connectedness. Regarding connectedness or cohesion, metrics were introduced by [27], being number of root classes, number of leaf classes, and average depth of inheritance tree of all leaf node. These metrics are not aimed at evaluating the quality of modules, but are rather general for ontologies. In light of this, Oh et. al present new metrics for cohesion to measure the strength of the relations in a module [20], and semantic dependencies were proposed in [5]. Ensan and Du's metrics in [5] use the notion of strong and moderate dependencies between entities. However, there are certain relations in an ontology that are neither strong nor moderate, such as intersections of classes.

3 Evaluation metrics

The evaluation metrics for modularity was compiled by studying existing literature on modularity. This resulted in 13 metrics from the literature, of which five were short of a metric for quantitative evaluation that have now been devised (indicated with an asterisk), and three new ones have been added (indicated with a double asterisk)[3].

Size The size metric is a fairly common metric as a modularity evaluation criterion [3, 4, 20, 22, 24]. Size refers to the number of entities in a module, $|M|$, which can be subdivided into number of classes, $|C|$, object properties $|OP|$, data properties $|DP|$, and individuals $|I|$. Size is calculated as follows: $Size(M) = |M| = |C| + |OP| + |DP| + |I|$. Note that it excludes the number of axioms, because that is considered a structural criterion.

Appropriateness of module size can be specified by mapping the size of an ontology module to some appropriateness values. Schlicht and Stuckenschmidt [24] propose an appropriate function to measure this, which ranges between 0 and 1, where a module with an optimal size has a value of 1. The function they propose is based on software design principles: since the optimal size of software modules is between 200-300 logical lines of software code, an axiom value of 250 would be the optimal size for an ontology, restricting the module to be between 0 and 500 axioms. The appropriateness equation is defined as follows [24], where x is the number of axioms in the module: $Appropriate(x) = \frac{1}{2} - \frac{1}{2}cos(x.\frac{\pi}{250})$.

Attribute richness is defined as the average number of attributes per class [25]; i.e., each class is defined by a number of axioms with properties describing describing it, which are referred to as attributes: $AR(M) = \frac{|att|}{|C|}$ where att is the number of attributes (OWL properties) of all entities and $|C|$ is the number of classes in the module.

[3] an earlier version of this section was presented at [14] and has now been updated with some corrections, refinements, and better descriptions.

Inheritance richness refers to how the knowledge is distributed across the ontology [25], such as with deep class hierarchies versus one with a flat or horizontal structure with few subclasses; this is calculated as follows: $IR_S(M) = \frac{\sum_{C_i \in C} |H^C(C_1, C_i)|}{|C|}$ where $|H^C(C_1, C_i)|$ is the number of subclasses ($C1$) for a class Ci and $|C|$ is the total number of classes in the ontology.

Cohesion refers to the extent to which entities in a module are related to each other. We use a metric defined in [20]:

$$Cohesion(M) = \begin{cases} \sum_{C_i \in M} \sum_{C_j \in M} \frac{SR(c_i, c_j)}{|M|(|M|-1)} & if |M| > 1 \\ 1 & otherwise \end{cases}$$

where $|M|$ is the number of entities in the module and $|M|(|M|-1)$ represents the number of possible relations between entities in M. The strength of relation for each entity is calculated based on a farness measure.

$$SR(c_i, c_j) = \begin{cases} \frac{1}{farness(i)} & if\ relations\ exist\ between\ c_i\ and\ c_j \\ 0 & otherwise \end{cases}$$

Redundancy has been defined as the duplication of axioms within a set of ontology modules [24]. When a large ontology is partitioned into smaller modules, there are sometimes modules that overlap with regard to shared knowledge. Thus axioms exist in more than one modules. Sometimes this is required for robustness or efficiency. However, these redundant axioms cause difficulty in maintaining the consistency of the modules when modules are to be updated. To measure redundancy in a set of modules, we use: $Redundancy = \frac{(\sum_{i=1}^{k} n_i) - n}{\sum_{i=1}^{k} n_i}$.

Correctness states that every axiom that exists in the module also exists in the original ontology and that nothing new should be added to the module [2, 4, 16, 22], i.e.: $Correctness(M) = M \subseteq O$.

Completeness A module is logically complete if the meaning of every entity is preserved as in the source ontology. The completeness property evaluates that for a given set of entities or signature, every axiom that is relevant to the entity as in the source ontology is captured in the module [2, 4, 16, 22]. $Completeness(M) = \sum_{i}^{n} Axioms(Entity_i(M)) \models Axioms(Entity_i(O))$.

Intra-module distance* d'Aquin et al. [4] introduce the notion of intra-module distance as the distance between entities in a module, which may be calculated by counting the number of relations in the shortest path from one entity to the other, for every entity in the module. The shortest path is calculated based on the entity hierarchy. Based on the description by d'Aquin et al., we refine this to measuring this distance by using Freeman's Farness value [6]. In the field of network centrality, Freeman's Farness value of a node is described as the sum of its distances to all other nodes in the network:

$$Intra\text{-}module\ distance(M) = \sum_{i}^{n} Farness(i) \tag{1}$$

with n the number of nodes in the module and the Farness value defined as $Farness(i) = \sum_{j}^{n} distance_{ij}$ The distance then is measured as the length of the shortest path between entities.

Inter-module distance* in a set of modules has been described as the number of modules that have to be considered to relate two entities [3, 4]. Based on this definition, we have created an equation to measure the inter-module distance of a network of modules.

$$IMD = \begin{cases} \sum\limits_{C_i, C_j \in (M_i,, M_n)} \frac{NM(C_i, C_j)}{|(M_i,.., M_n)|(|(M_i,.., M_n)|-1)} & |(M_i, .., M_n)| > 1 \\ 1 & otherwise \end{cases} \quad (2)$$

where $NM(C_i, C_j)$ is the number of modules to consider to relate entities i and j and $|(M_i, .., M_n)|(|(M_i, .., M_n)| - 1)$ represents the number of possible relations between entities in a set of modules $(M_i, ., M_n)$.

Coupling* has been described as a measure of the degree of interdependence of a module [7, 19–21]. The coupling value is high if entities in a module have strong relations to entities in other modules. This also means that it will be difficult to update such modules independently because they affect other modules in the system. To measure the coupling of a module, we define our own measure as a ratio of the number of external links (axioms) between a module M_i and M_j, NEL_{M_i,M_j} for n modules in a system to every possible external link between a module M_i and M_j in a system.

$$Coupling(M_i) = \begin{cases} \sum\limits_{i=0}^{n} \sum\limits_{\substack{j=0 \\ i \neq j}}^{n} \frac{NEL_{M_i,M_j}}{|M_i||M_j|} & NEL_{M_i,M_j} > 0 \\ 0 & otherwise \end{cases} \quad (3)$$

where $|M_i|$ is the number of entities in the current module and $|M_j|$ is the number of entities in a related module in the set of n modules. External links in ontology modules depend on what linking language is used.

Encapsulation* d'Aquin et al. mention encapsulation with the notion that "a module can be easily exchanged for another, or internally modified, without side-effects on the application can be a good indication of the quality of the module" [4]. This general idea seems potentially useful for semantic interoperability. There are thus two components to d'Aquin et al.'s encapsulation:

- 'Swappability' of a module, which increases with fewer links to entities in another module in an ontology network; e.g., one can interchange their domain ontologies between foundational ontologies using the SUGOI tool [13].
- Casting it into a measure of knowledge preservation within the given module.

We have designed an equation to calculate the encapsulation of a module in a given a set of modules. For a module, with $n - 1$ related , this is measured using the number of axioms in the given module $|Ax_i|$ and the number of axioms

that overlap between the given module and related modules, $|Ax_{ij}|$.

$$Encapsulation(M\_i) = 1 - \frac{\sum_{j=1}^{n-1} \frac{|Ax_{ij}|}{|Ax_i|}}{n} \tag{4}$$

Encapsulation values in modules that are equal or close to 1 indicates a good encapsulation value; all or most of the knowledge has been encapsulated and privacy has been completely preserved. Conversely, values that are equal to or close to 0 indicates a poor encapsulation value; none or very little of the knowledge has been encapsulated and privacy has not been preserved.

Independence* Independence evaluates whether a module is self-contained and can be updated and reused separately. In this way, ontology modules can evolve independently. Thus, the semantics of the entire ontology could change without the need for all the modules to be changed. For instance, for the set of Gist foundational ontology modules [17], if information about physical things need to be updated, the relevant module gistPhysicalThing could be updated without needing to alter the remaining modules. In order to determine whether a module is independent, we use two metrics, i.e., the encapsulation and the coupling measure. A module is set to be independent if it has an encapsulation value of 1 and a coupling value of 0. This can be checked using the following code snippet.

$$Ind(M\_i) = \begin{cases} true & Encapsulation(M\_i) = 1 \text{ and } Coupling(M\_i) = 0 \\ false & otherwise \end{cases} \tag{5}$$

where $|M_i|$ is the number of entities in the current module and $|M_j|$ is the number of entities in a related module in the set of n modules.

Relative size** can be defined as the size of the module—i.e., number of classes, properties, and individuals—compared to the original ontology. The relative size of a module strongly influences the result of the module on tasks such as reasoning and maintenance, for if the module extracted is nearly the same as the original one, then not substantial optimisation will be obtained. To compute this, we use the ratio of the size of the module M (i.e., $|M|$) and the original (source) ontology O (i.e., $|O|$):

$$Relative\ size = \frac{|M|}{|O|} \tag{6}$$

Thus, a lower value (between 0 and 1) is better.

Atomic Size** The notion of atoms within ontology modules was first introduced in [26], who define it as a group of axioms within an ontology that have dependencies between each other. Based on the findings from the study, that it is possible to modularise an ontology using atomic decomposition as a method, we propose to measure the size of atoms in ontologies. We define the atomic size as the average size of a group of inter-dependent axioms in a module, and

formulate an equation to measure the atomic size of a module by using the sum of all the atoms present in the module, and the size of the ontology.

$$Atomic\ Size(M) = \sum_{i}^{n} \frac{Atom_i}{|M|} \tag{7}$$

Relative Intra-module distance** can be defined as the difference between distances of entities in a module M to a source ontology O. This difference would reveal if the overall distance between the entities in the module has been reduced, and by how many distance units. This is useful in comparing the difference in module size; whether the technique reduces the size considerably. To compare the distances of the original ontology, we compute the farness values for the subset of nodes that exist in a module, which is used to calculate the intra-module distance of the original ontology, and is defined as follows:

$$Relative\ intra\text{-}module\ distance(M) = \frac{Intra\text{-}module\ distance(O)}{Intra\text{-}module\ distance(M)} \tag{8}$$

4 Implementation and evaluation

We have created a Tool for Ontology Modularity Metrics (TOMM) to evaluate ontology modules. TOMM allows one to upload a module or set of related ontology modules, together with an original ontology (if it exists), and then it computes metrics for the module/s. A screenshot of TOMM's interface is shown in Fig. 1. The metrics are saved as a text file on the user's computer.

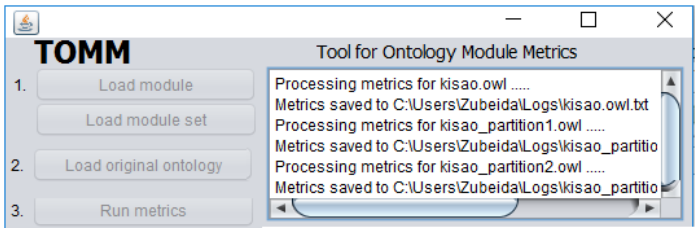

Fig. 1. The interface of TOMM.

4.1 Experimental evaluation

The purpose of the experiment is to evaluate modules with a set of metrics. We expect that the results will determine how the metrics of a module relate to other factors, such as technique to create them.

Table 1. Averages for a subset of TOMM's metrics; $|T|$ = number of module types, approp. = appropriateness, IMD = intra module distance.

	$\|T\|$	**Relative size**	**Atomic size**	**Approp.**	**Relative IMD**	**Attribute richness**	**Inheritance richness**
T1	13	0.02	5.50	0.34	20.69	0.83	1.48
T2	42	-	5.31	0.64	-	1.45	2.37
T3	7	0.90	6.31	0.11	1.00	0.84	2.30
T4	3	0.02	5.00	0.47	63.66	3.61	1.79
T5	2	0.30	7.20	0.61	1.04	0.87	2.45
T6	10	0.17	2.99	0.30	0.00	0.10	54.32
T7	90	0.01	1.00	0.007	0.00	0.50	1.19
T8	4	0.56	3.64	-	1.02	0.71	3.15
T9	1	1.00	2.89	-	0.00	0.00	2.83
T10	1	0.56	4.21	0.99	1.03	0.00	3.06
T11	3	0.49	3.77	0.89	1.00	0.58	2.44
T12	3	0.42	5.87	0.02	2.17	1.05	2.89
T13	6	1.00	4.33	0.38	1.00	0.73	2.72
T14	1	0.97	5.65	-	1.00	1.78	3.04

Materials and methods The method for the experiment is straightforward: 1) take a set of ontology modules ; 2) run the TOMM tool for each module; 3) conduct an analysis from the evaluation results for each module. In order to determine which metrics can be used to evaluate which module types, we need to determine how to interpret the values for each metric, which are as follows:

– correctness, completeness and independence are measured as true/false;
– size, no. of axioms, atomic size, intra-module distance. relative intra-module distance, attribute richness, and inheritance richness are measured on a numerical range;
– relative size, appropriateness, cohesion, encapsulation, coupling, and redundancy are measured on a 4-point scale of small (0-0.25), medium (0.25-0.5), moderate (0.51-0.75), and large (0.75-1).

The materials used for the experiment were as follows: Protégé v4.3 [8], TOMM, and a set of ontology modules that serve as the training set. Khan and Keet's set [15] was reused, which contains 189 ontology modules that were collected from ontology repositories and as referenced in the literature. This set contains modules of 14 different types, which are summarised in the appendix. All the test files used for this experimental evaluation can be downloaded from `www.thezfiles.co.za/Modules/testfiles.zip`.

Results We ran TOMM for each of the 189 modules and metrics were successfully generated for 188. Due to space limits we include only Table 1 with average values for a subset of the metrics and highlight the notable aspects of the results here; the remaining metric tables are available online together with the test files.

For size, T7 (ontology matching) modules are very small, only 2% compared to the original ontology. T2 (subject domain) could not be evaluated with the

relative size metric as there were no original ontologies. T13 (expressiveness sublanguage) is as large as the original ontology. For appropriateness, T10 (entity type abstraction) is the most appropriate at 0.99, meaning that most of the modules have between 200-300 axioms. The relative intra-module distance values determine by how many units (paths between entities) the module has been reduced. T4 (locality) modules were reduced with a high value by 63.65 units followed by T1 (ontology design patterns) by 20.69 units.

The T4 (locality), T8 (optimal reasoning), T9 (axiom abstraction), and T10 (entity type abstraction) modules all hold the correctness metrics; every axiom that exists in the module also exists in the source ontology and nothing new had been added. T1 (ontology design pattern) modules are the only set that all hold the completeness metric; the meaning of every entity in the module is preserved as in the source ontology. For attribute richness, T4 (locality) modules were the richest with a value of 3.61; these modules have on average 3.61 attributes per class. For inheritance richness, T6 (domain coverage) modules had a large value of 54.32 indicating many subclasses per class.

The information hiding and relational criteria only apply to module sets, T2 (subject domain), T6 (domain coverage), T7 (ontology matching), and T8 (optimal reasoning). For encapsulation, T7 (ontology matching) modules had a high value of 1; the knowledge is preserved in the individual modules and they can be changed individually without affecting the other modules in the set. For coupling, most of the modules had 0 values (no links to other modules in the set). The T7 (ontology matching) modules are independent; they are self-contained and also do not contain links to other modules in the set. The experiment uncovered which metrics fare well with which module type as discussed in this section and included in Fig. 2, where for each module type, the metrics and values that fare well with it are stated in bold font.

It is also worthwhile to check whether the techniques used for modularisation have an effect on the quality of the module. For the set of 189 modules in the set, there were four techniques used to generate them: graph partitioning, locality-based modularisation, *a priori* modularisation, and manual methods. The modules that were generated via graph partitioning measured well for the following criteria: relative size (small), encapsulation (large), coupling (small), redundancy (small). The modules generated with locality-based modularity performed well for correctness (all true). *a priori* modules all performed well for encapsulation (large), coupling (small), and redundancy (small). There was no link between the metrics returned by the modules generated by manual methods; all the results differed.

Use-cases We selected two existing cases of ontology modularisation to evaluate TOMM and the resulting metrics, which are modules not in the training set.

Example 1 (QUDT ontology modules). The Quantities, Units, Dimensions and Data Types (QUDT) ontology modules are a set of modules about science terminology for representing physical quantities, units of measure, and their dimensions [10]. According to the framework for ontology modularity, these modules

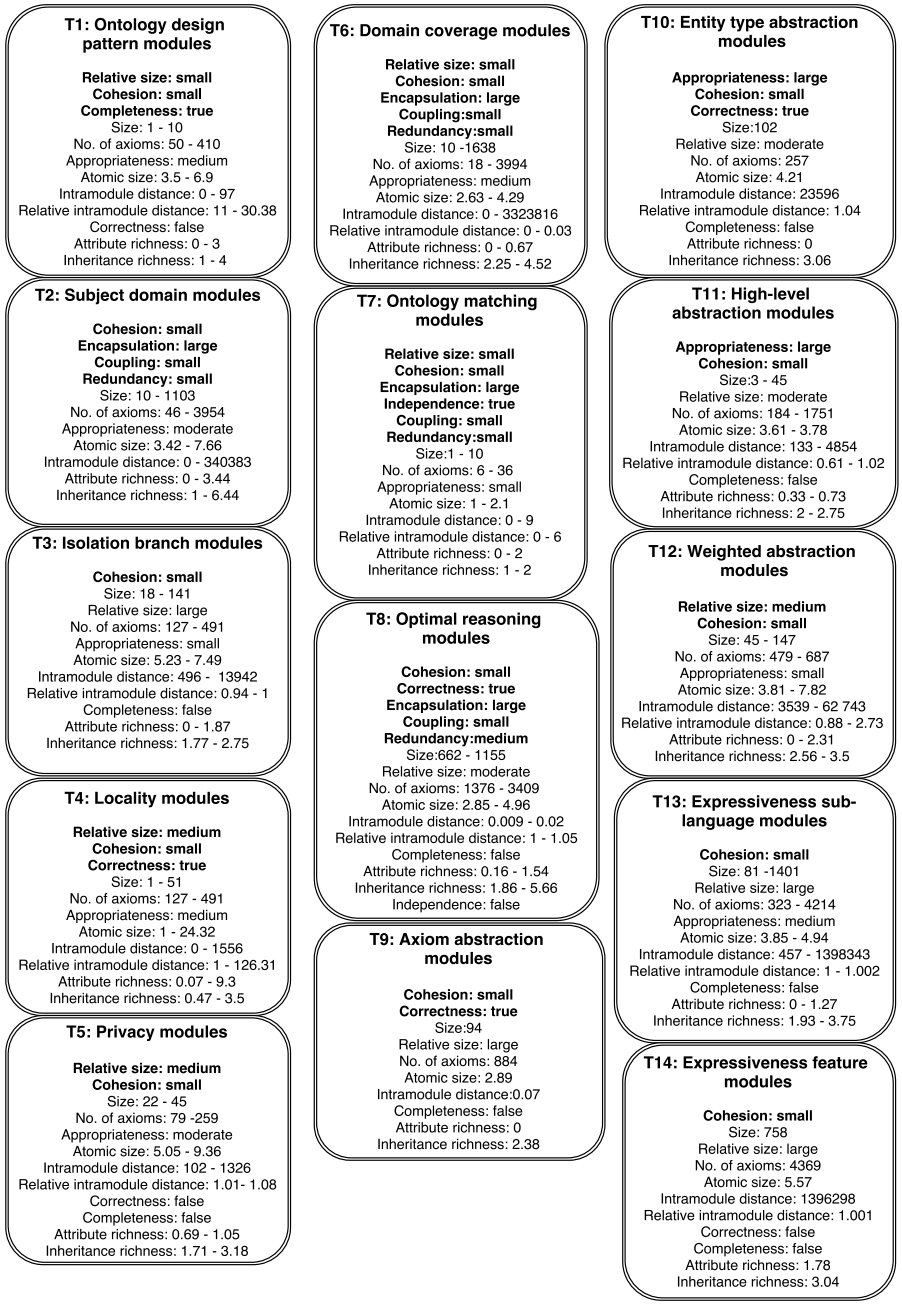

Fig. 2. The set of metrics that can be measured for each module type. Metrics and values in bold font are those which evaluate well for a module type.

are of type T2: Subject domain modules. The modules fare well for 3 out of the 4 metrics that are expected of T2: Subject domain modules; the cohesion is small, encapsulation is large, and coupling is small (see Table 2). The redundancy of the QUDT modules is 0.50 which is moderate, as opposed to an expected small value. For the metrics that are measured by their numerical values only, i.e., atomic size, attribute richness, etc., the metrics are within the expected ranges summarised in Fig. 2. Thus according to the metrics, the QUDT ontology modules are of good quality as subject domain modules.

Table 2. The metrics for the QUDT ontology modules generated by TOMM; approp = appropriateness, encap. = encapsulation, avg. = average, med. = median.

	Structural criteria					
	Size	**Atomic size**	**No. of axioms**	**Approp.**	**Intra module distance**	**Cohesion**
Avg.	595.38	5.71	3112.00	0.91	8577.25	0.008
Med.	479.00	3.70	1443.50	0.91	86.50	0.003
	Richness criteria		**Information hiding criteria**		**Relational criteria**	
	AR	**IR**	**Encap.**	**Coupling**	**Independence**	**Redundancy**
Avg.	1.69	1.89	0.99	0	25% true	0.50
Med.	1.40	1.84	0.99	0	75% false	0.50

Example 2 (The Pescado Ontology). The Pescado ontology contains knowledge about the environment, such as meteorological conditions and phenomena, air quality, and disease information [23]. The PescadoDisease module is a subset of information only about diseases, so it is a locality module (Type T4). The module fares well for the cohesion metric, which is small, the appropriateness value (being medium), the correctness metric (true), and for all those metrics measured by numerical ranges too, according to the expected values of Figure 2. The only metric that differs is relative size: the PescadoDisease module is small compared to the experimental data where locality modules were medium.

Using TOMM and the use-cases, we were able to evaluate the quality of ontology modules. QUDT and Pescado are different types of modules and therefore different values are expected for their metrics. With both modules, for all their metrics except one, the values generated by TOMM are as expected for their types; this indicates that the modules are of 'good' quality.

5 Discussion

The list of module metrics that was compiled is a first step in solving the problems regarding the evaluation of ontology modules, and, following from that, knowing how to create a good module. The metrics that are programmed into

Table 3. The metrics for the Pescado disease ontology generated by TOMM; app = appropriateness.

Structural criteria							
Size	**Atomic size**	**No. of axioms**	**App.**	**Intra module distance**	**Cohesion**	**Relative size**	**Relative intra module distance**
39.00	3.10	128.00	0.51	158.00	0.16	0.03	10.61
Richness criteria				**Logical criteria**			
Attribute richness		**Inheritance richness**		**Correctness**		**Completeness**	
0.00		1.67		True		True	

TOMM allow one to evaluate ontology modules using different metrics such as logical (correctness), structural (relational size), relational (coupling), and so on. Of all the metrics, it was not feasible to include the inter-module distance metric in the program, because these modules were linked using ε-connections, which could not be recognised by the OWL API that was used to develop TOMM. Also, in testing, the 'FMA_subset' module (from T12: weighted abstraction modules) was too large for TOMM to process due to insufficient Java heap space size and increasing the parameters caused the machine to crash. We are looking at running TOMM on a High-Performance Computing Cluster in the future.

We have evaluated modules with TOMM, and analysed their metrics. The results reveal which metrics fare well with which module type, as displayed in Figure 2; e.g., T1 (ontology design patterns) modules are relatively small compared to the original ontology, and the completeness value is true. For T3, T13, and T14 modules, there is limited associations between them and the metrics. The analysis reveals that they all only fare well for the cohesion metric; all the sets of modules fare well for the cohesion metric.

For the bulk of the modules, T3, T5, T11, T12, T13, and T14 provide good results for structural metrics. Modules of type T1, T4, T9, and T10 have good results for both structural, and logical metrics. Modules of type T2, T6, and T7 have good results for structural, information hiding, and relational metrics, and T8 type of modules have good results for some criteria, structural, logical, relational, and information hiding. Richness criteria only returns a range of numerical values which cannot be mapped to rate values such as small, medium, etc., hence it is unclear what ideal values for such criteria are. Thus, using TOMM to evaluate a module, a user is able to determine whether the module is of 'good' quality. Our approach of evaluating whether a module is of 'good' quality heavily depends on the data from existing modules used in this experiment. The reason for this approach is to offer the developer a practical solution for evaluating modules in Semantic Web applications.

From the assessment on any relation between modularisation technique and ontology module quality metrics, it exhibited a link between the graph partitioning, locality-based, and *a priori* techniques and the metrics; there were certain metrics that were associated with each of the respective techniques. There were

four metrics associated with graph partitioning, one with locality, and three with *a priori* techniques. Unsurprisingly, there does not seem to be any clear association between manual modularisation technique and any of the metrics. Perhaps an in-depth qualitative assessment of the manually created modules may reveal what is going on exactly.

The use-case evaluation with the QUDT (of type subject domain modules) and the Pescado-disease modules (of type locality) were promising, showing good modules for their respective types. Others may not fare as well, which time may tell. Most ontology modules we could find are already included in the test set, so that will depend on the modules that are being developed, which, however, can avail of the results presented here to exactly avoid creating a 'bad' module.

6 Conclusion

Five new modularity metrics with measures and three new measures for existing metrics were proposed, making the total to 16 ontology module evaluation metrics. They have been implemented in the TOMM tool to enable scaling up of module evaluation. Our evaluation carried out with 189 modules revealed which metrics work well with which types of modules. This is displayed in bold font in Figure 2; for each of the 14 module types, the metrics that fare well with them together with the expected values are displayed. It is now possible for an ontology developer to evaluate the quality of a module/set of modules by first classifying its type using the framework for ontology modularity, and then generating its metrics using the TOMM metrics tool. Ontology developers are then able to determine whether their ontology module is of 'good quality' based on comparing the module's metrics to what is expected in Figure 2.

For future work, we aim to achieve more insight into module evaluation by linking the module evaluation metrics to other characteristics of the modularity framework such as use-cases and properties, to reveal more dependencies. It is also worthwhile to apply the tool to ontology design patterns towards improving pattern quality.

References

1. Borgo, S.: Goals of modularity: A voice from the foundational viewpoint. In: Fifth International Workshop on Modular Ontologies (WOMO'2011). Frontiers in Artificial Intelligence and Applications, vol. 230, pp. 1–6. IOS Press (2011), ljubljana, Slovenia, August
2. Cuenca Grau, B., Parsia, B., Sirin, E., Kalyanpur, A.: Modularity and Web Ontologies. In: 10th International Conference on Principles of Knowledge Representation and Reasoning (KR'06). pp. 198–209. AAAI Press (2006), June 2-5, Lake District, United Kingdom
3. d'Aquin, M., Schlicht, A., Stuckenschmidt, H., Sabou, M.: Ontology modularization for knowledge selection: Experiments and evaluations. In: 18th International Conference on Database and Expert Systems Applications (DEXA'07). LNCS, vol. 4653, pp. 874–883. Springer (2007), regensburg, Germany, September 3-7, 2007

4. d'Aquin, M., Schlicht, A., Stuckenschmidt, H., Sabou, M.: Criteria and evaluation for ontology modularization techniques. In: Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization, LNCS, vol. 5445, pp. 67–89. Springer (2009)
5. Ensan, F., Du, W.: A semantic metrics suite for evaluating modular ontologies. Information Systems 38(5), 745–770 (2013)
6. Freeman, L.C.: Centrality in social networks conceptual clarification. Social Networks 1(3), 215–239 (1978)
7. García, J., Peñalvo, F.J.G., Therón, R.: A survey on ontology metrics. In: Third World Summit on the Knowledge Society, (WSKS'10). CCIS, vol. 111, pp. 22–27. Springer (2010), corfu, Greece, September 22-24
8. Gennari, J.H., Musen, M.A., Fergerson, R.W., Grosso, W.E., Crubézy, M., Eriksson, H., Noy, N.F., Tu, S.W.: The evolution of Protégé: an environment for knowledge-based systems development. International Journal of Human-Computer Studies 58(1), 89–123 (2003)
9. Grau, B.C., Horrocks, I., Kazakov, Y., Sattler, U.: Modular reuse of ontologies: Theory and practice. Journal of Artificial Intelligence Research 31, 273–318 (2008)
10. Hodgson, R., Keller, P.J.: QUDT-quantities, units, dimensions and data types in OWL and XML. http://www.qudt.org (2011), online (September 2011)
11. Kalyanpur, A., Parsia, B., Sirin, E., Cuenca Grau, B., Hendler, J.A.: Swoop: A Web Ontology Editing Browser. Journal of Web Semantics 4(2), 144–153 (2006)
12. Khan, Z., Keet, C.M.: The foundational ontology library ROMULUS. In: Third International Conference on Model & Data Engineering (MEDI'13). LNCS, vol. 8216, pp. 200–211. Springer (2013), september 25-27, 2013, Amantea, Italy
13. Khan, Z., Keet, C.M.: Feasibility of automated foundational ontology interchangeability. In: 19th International Conference on Knowledge Engineering and Knowledge Management (EKAW'14). LNAI, vol. 8876, pp. 225–237. Springer (2014), 24 - 28 November 2014, Linköping, Sweden
14. Khan, Z.C.: Evaluation metrics in ontology modules. In: 29th International Workshop on Description Logics (DL'16). CEUR Workshop Proc., vol. 1577. CEUR-WS.org (2016), 22-25 April 2016, Cape Town, South Africa
15. Khan, Z.C., Keet, C.M.: An empirically-based framework for ontology modularisation. Applied Ontology 10(3-4), 171–195 (2015)
16. Loebe, F.: Requirements for logical modules. In: First International Workshop on Modular Ontologies (WoMO'06). CEUR Workshop Proc., vol. 232. CEUR-WS.org (2006), november 5, Athens, Georgia, USA
17. McComb, D.: Gist: The minimalist upper ontology. Semantic Technology Conference (2010), june 21-25 2010, San Francisco, CA
18. Noy, N.F., Musen, M.A.: Specifying Ontology Views by Traversal. In: Third International Semantic Web Conference (ISWC'04). LNCS, vol. 3298, pp. 713–725. Springer (2004), nov 7-11, Hiroshima, Japan
19. Oh, S., Ahn, J.: Ontology module metrics. In: International Conference on e-Business Engineering, (ICEBE'09). pp. 11–18. IEEE Computer Society (2009), macau, China, 21-23 October
20. Oh, S., Yeom, H.Y., Ahn, J.: Cohesion and coupling metrics for ontology modules. Information Technology and Management 12(2), 81–96 (2011)
21. Orme, A.M., Yao, H., Etzkorn, L.H.: Coupling metrics for ontology-based systems. IEEE Software 23(2), 102–108 (2006)
22. Pathak, J., Johnson, T.M., Chute, C.G.: Survey of modular ontology techniques and their applications in the biomedical domain. Integrated Computer-Aided Engineering 16(3), 225–242 (2009)

23. Rospocher, M.: An ontology for personalized environmental decision support. In: Formal Ontology in Information Systems FOIS '14. pp. 421–426 (2014), september, 22-25, 2014, Rio de Janeiro, Brazil
24. Schlicht, A., Stuckenschmidt, H.: Towards structural criteria for ontology modularization. In: First International Workshop on Modular Ontologies, (WoMO'06). CEUR Workshop Proc., vol. 232. CEUR-WS.org (2006), november 5, 2006, Athens, Georgia, USA
25. Tartir, S., Arpinar, I.B., Moore, M., Sheth, A.P., Aleman-Meza, B.: OntoQA: Metric-based ontology quality analysis. IEEE Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources 9 (2005)
26. Vescovo, C.D.: The modular structure of an ontology: Atomic decomposition towards applications. In: 24th International Workshop on Description Logics (DL 2011). CEUR Workshop Proc., vol. 745. CEUR-WS.org (2011), barcelona, Spain, July 13-16
27. Yao, H., Orme, A.M., Etzkorn, L.: Cohesion metrics for ontology design and application. Journal of Computer science 1(1), 107 (2005)

Appendix: summarised types of ontology modules

T1 *Ontology design pattern modules* An ontology is modularised by identifying a part of the ontology for general reuse.

T2 *Subject domain modules* A large domain is divided by subdomains present in the ontology.

T3 *Isolation branch modules* A subset of entities from an ontology is extracted but entities with weak dependencies to the signature are not to be included in the module.

T4 *Locality modules* A subset of entities from an ontology is extracted, including all entities that are dependent on the subset.

T5 *Privacy modules* Some information is hidden from an ontology.

T6 *Domain coverage modules* A large ontology is partitioned by its graphical structure and placement of entities in the taxonomy.

T7 *Ontology matching modules* An ontology is modularised for ontology matching into disjoint modules so that there is no repetition of entities.

T8 *Optimal reasoning modules* An ontology is split into smaller modules to aid in overall reasoning over the ontology.

T9 *Axiom abstraction modules* An ontology is modularised to have fewer axioms, to decrease the horizontal structure of the ontology.

T10 *Entity type abstraction modules* An ontology is modularised by removing a certain type of entity e.g., data properties or object properties.

T11 *High-level abstraction modules* An ontology is modularised by removing lower-level classes and only keeping higher-level classes.

T12 *Weighted abstraction modules* An ontology is modularised by a weighting decided by the developer.

T13 *Expressiveness sub-language modules* An ontology is modularised by using a sub-language of a core ontology language.

T14 *Expressiveness feature modules* An ontology is modularised by using limited language features.